


Instrukcja wgrania oprogramowanie do modułu ESP32 DEVKIT

## 1. Instalacja środowiska ArduinoIDE.

<https://www.arduino.cc/en/software>

Klikamy po prawej odpowiedni link, dla Windows polecam „Win 7 and newer”

## Downloads



**Arduino IDE 1.8.19**

The open-source Arduino Software (IDE) makes it easy to write code and upload it to the board. This software can be used with any Arduino board.

Refer to the **Getting Started** page for Installation instructions.

**SOURCE CODE**

Active development of the Arduino software is **hosted by GitHub**. See the instructions for **building the code**. Latest release source code archives are available **here**. The archives are PGP-signed so they can be verified using **this** gpg key.

**DOWNLOAD OPTIONS**

- Windows** Win 7 and newer
- Windows** ZIP file
- Windows app** Win 8.1 or 10 **Get**
- Linux** 32 bits
- Linux** 64 bits
- Linux** ARM 32 bits
- Linux** ARM 64 bits
- Mac OS X** 10.10 or newer

Release Notes

Checksums (sha512)

Następnie opcja „Just download”



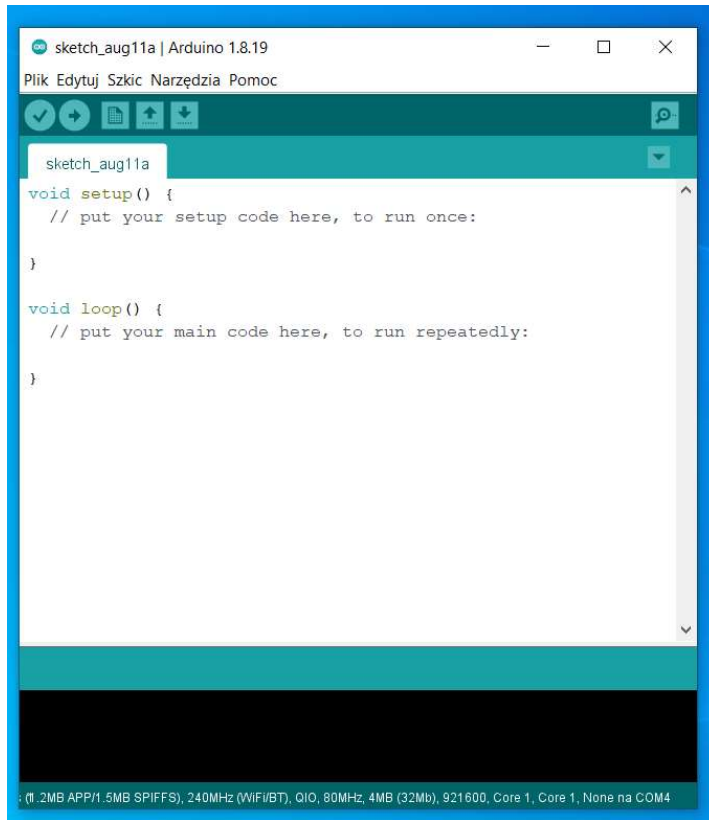
## Support the Arduino IDE

Since the release 1.x release in March 2015, the Arduino IDE has been downloaded **63 968 565** times — impressive! Help its development with a donation.

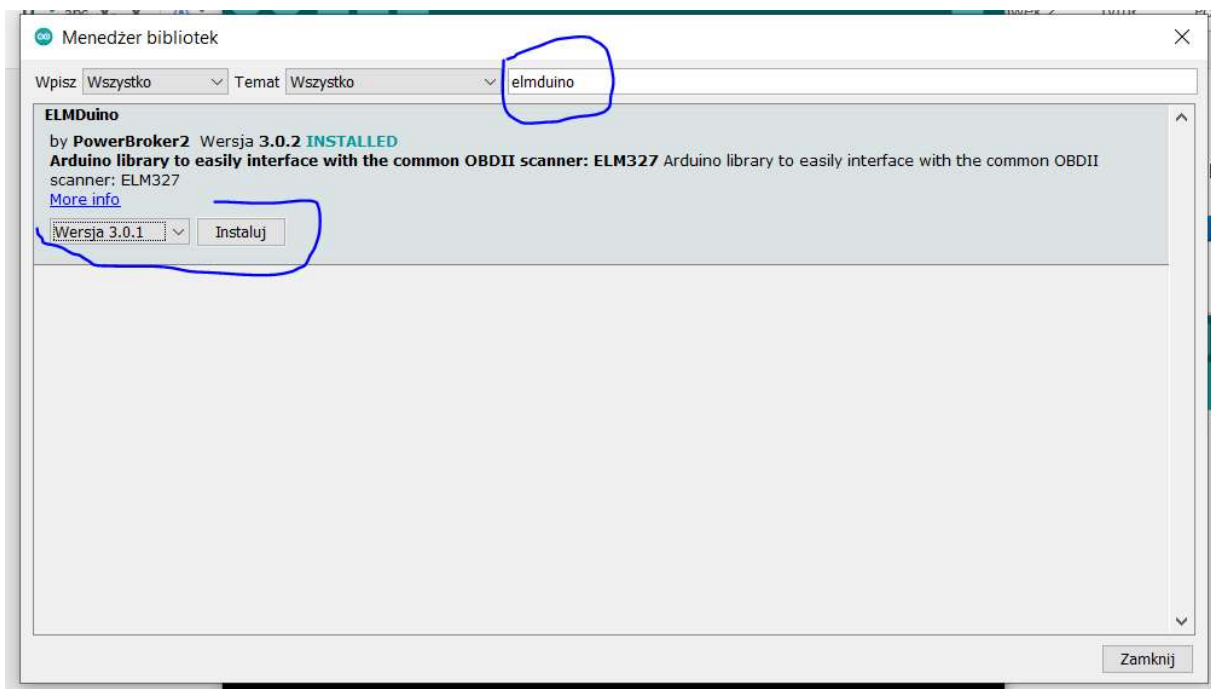
**\$3** **\$5** **\$10** **\$25** **\$50** **Other**

**JUST DOWNLOAD** **CONTRIBUTE & DOWNLOAD**

Instalujemy odpowiednio oprogramowanie, po uruchomieniu powinniśmy ujrzeć taki widok.

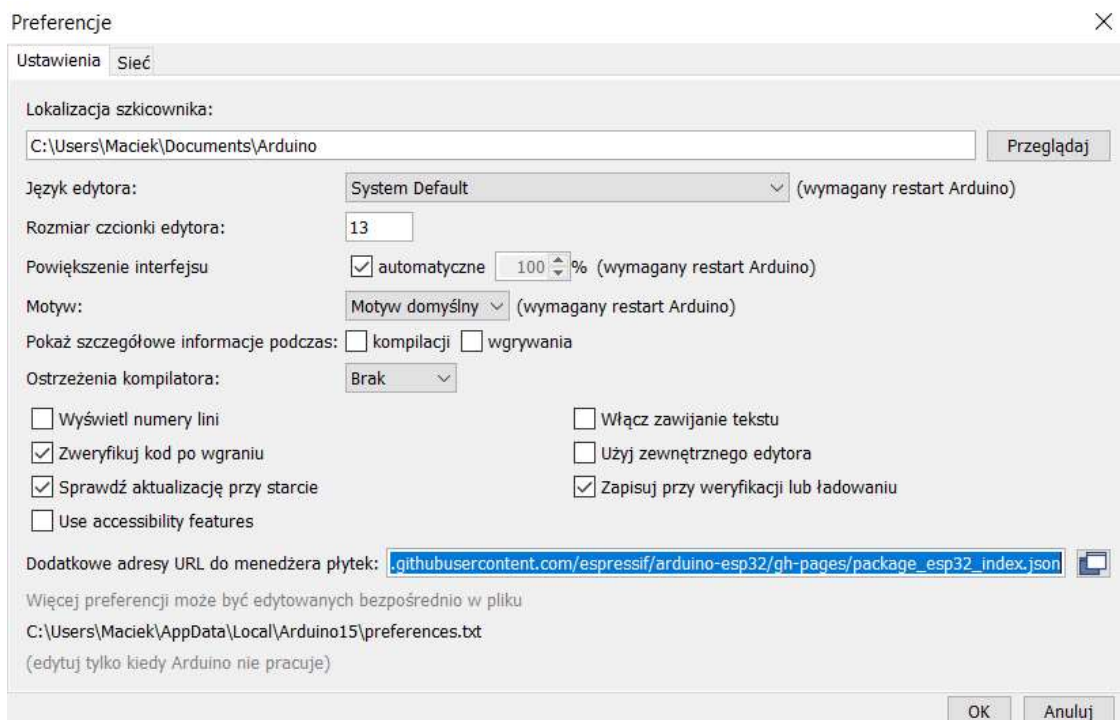


Klikamy „Narzędzia -> Zarządzaj bibliotekami”



W górnej wyszukiwarce wpisujemy „elmduino”, następnie wybieramy wersję 3.0.2 i klikamy „Instaluj” (u mnie widoczna jest wersja 3.0.1 bo mam już zainstalowaną 3.0.2). Po instalacji należy zweryfikować czy wersja 3.0.2 jest zainstalowana.

W kolejnym kroku klikamy „Plik -> Preferencje”

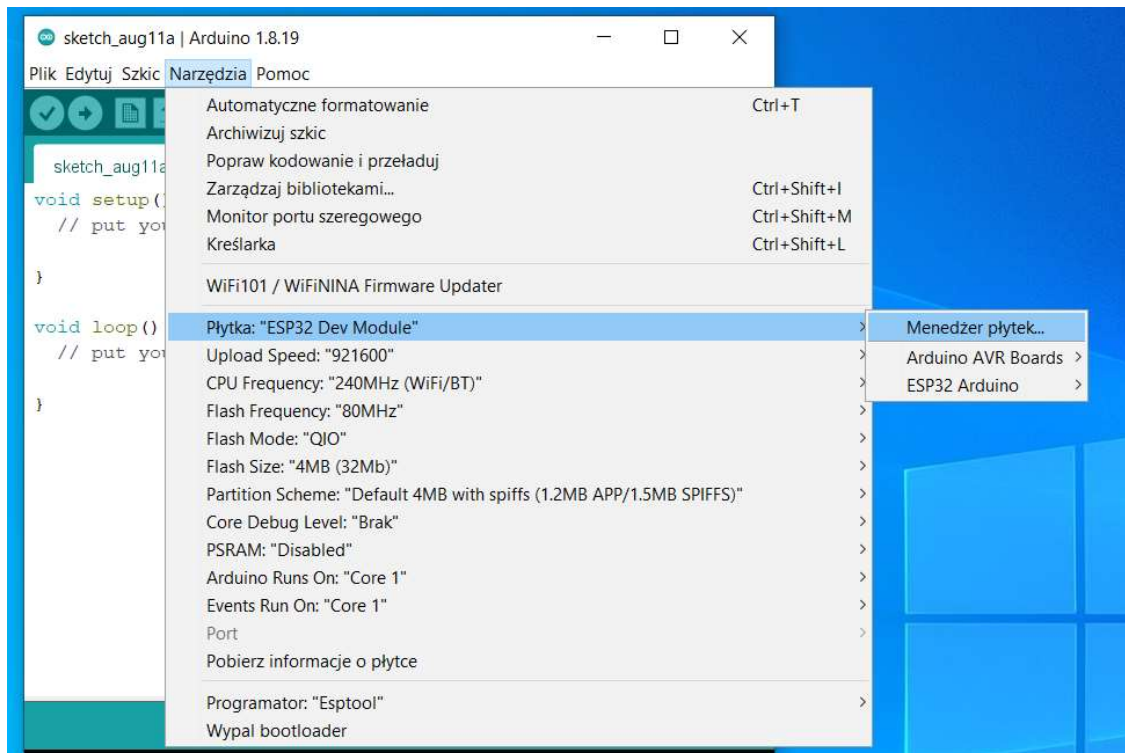


W miejscu „Dodatkowe adresy...” wklejamy poniższy link:

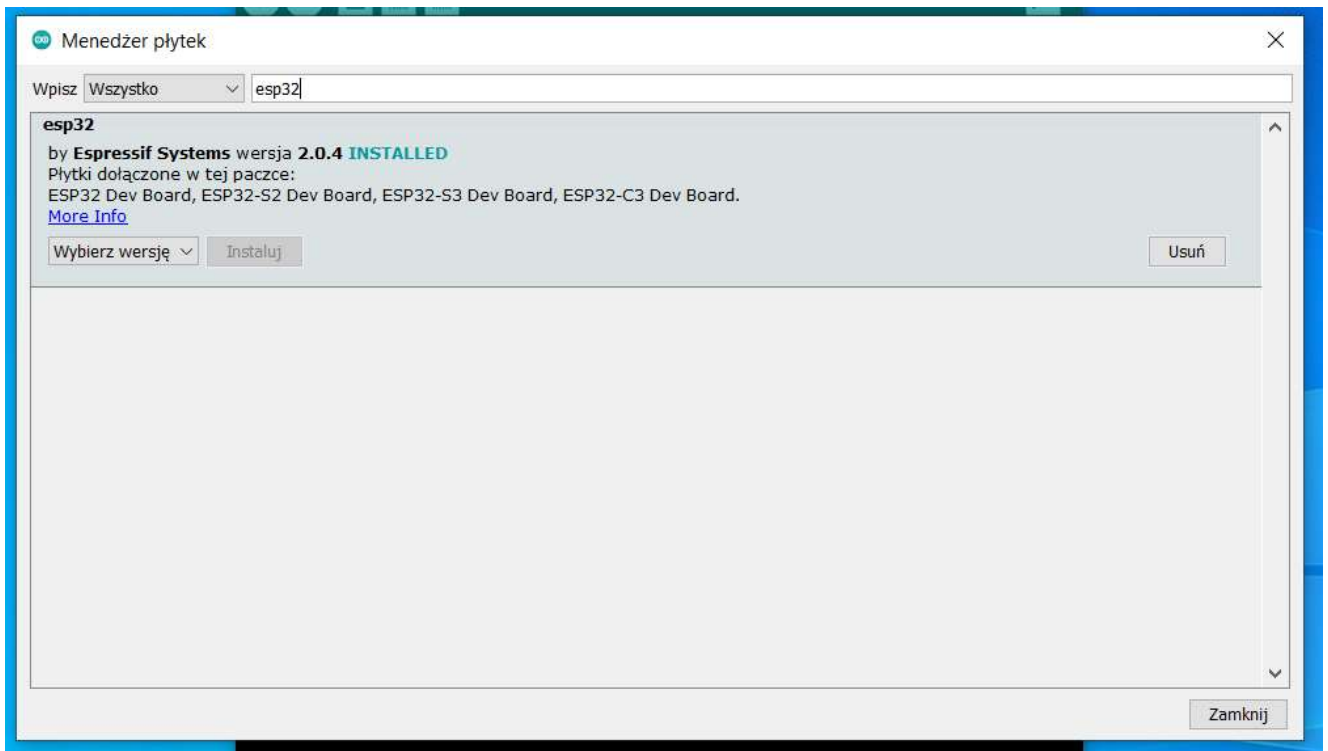
[https://raw.githubusercontent.com/espressif/arduino-esp32/gh-pages/package\\_esp32\\_index.json](https://raw.githubusercontent.com/espressif/arduino-esp32/gh-pages/package_esp32_index.json)

## Następnie instalujemy bibliotekę płytki ESP

Klikamy „Narzędzia -> Płytki -> Menadżer płytek”

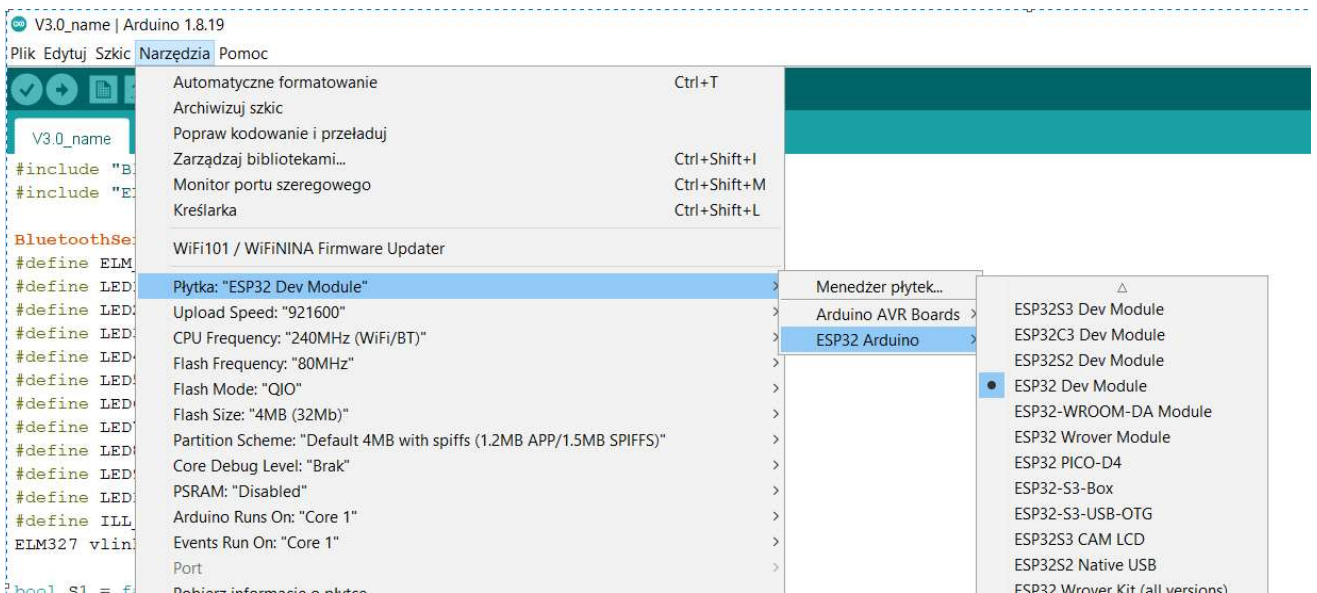


Po wejściu do menedżera wpisujemy „ESP32”, wybieramy 2.0.4 i klikamy instaluj. Po instalacji należy zweryfikować czy wersja 2.0.4 jest zainstalowana.



Zamykamy środowisko, uruchamiamy ponownie.

Po uruchomieniu wybieramy odpowiednią płytkę „Narzędzia -> Płytki -> ESP32 Arduino -> ESP32 Dev Module”

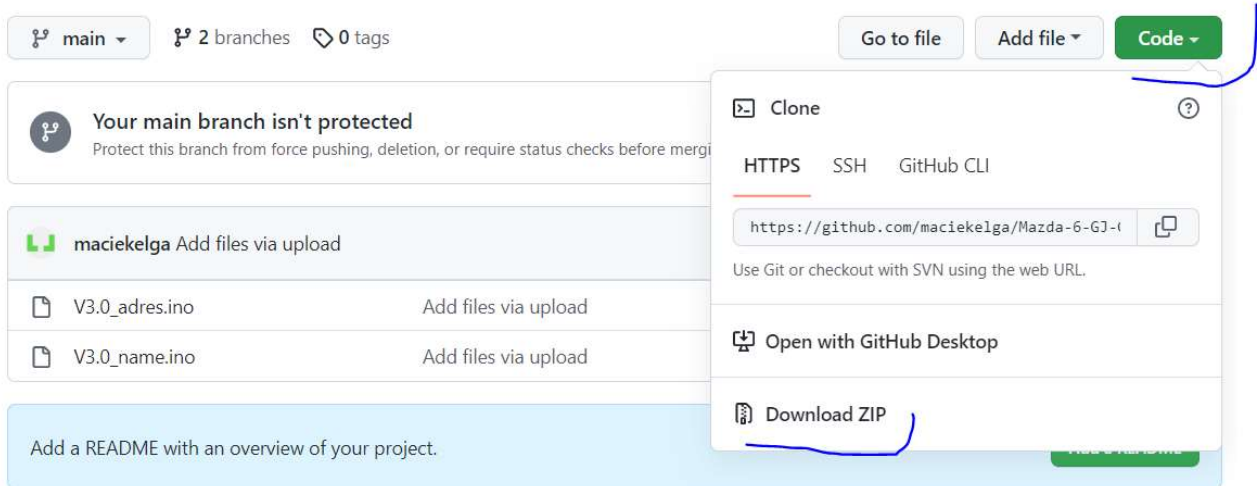


## 2. Pobieranie oraz przygotowanie kodu do wgrania.

Pobieramy paczkę z najnowszą wersją oprogramowania z poniższego linku

<https://github.com/maciekelga/Mazda-6-GJ-GL-DPF>

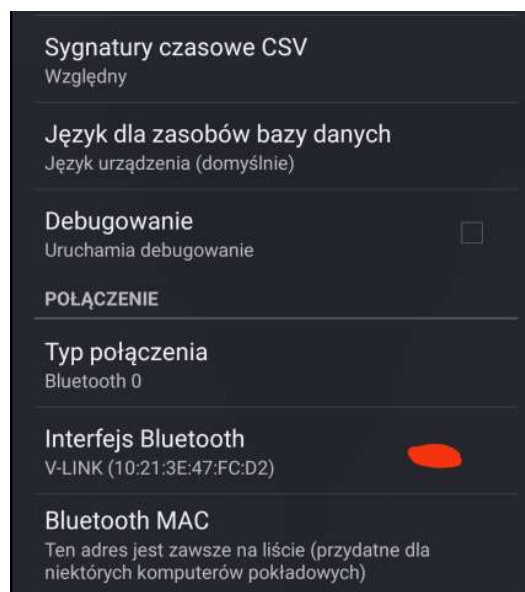
Klikamy w prawym rogu w „Code” następnie „Download ZIP”



W paczce znajdziemy dwie wersje programu, jeśli znamy MAC adres interfejsu BT otwieramy ten z „adres” w nazwie, jeśli nie znamy otwieramy „nazwa”. Otwieramy odpowiedni plik z poziomu środowiska ArduinoIDE klikając „Plik -> Otwórz”. Rekomenduje użycie wersji z MAC adres, choć obie działają poprawnie.

Następnie ujrzymy kod programu w którym musimy uzupełnić kilka rzeczy

- Autoryzacja na podstawie MAC adresu interfejsu BT - Adres można odczytać z poziomu Forscana Lite na Androidzie (Ustawienia -> Interfejs Bluetooth).



Odszukujemy kod programu zaznaczony poniżej i uzupełniamy nasz MAC adres zgodnie z poniższym przykładem

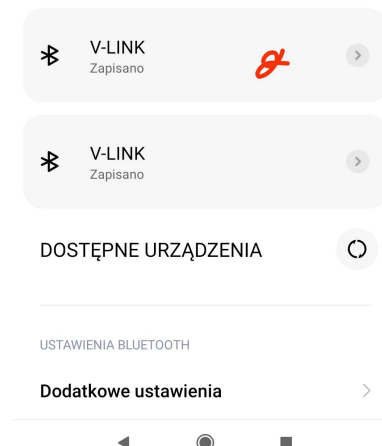
```
float PM_ACC_calc_abs = 0; //zmienna wartosci absolutnej roznicy wartosci aktualnej i z chwili poprzedniej

unsigned long previousMillis = 0; //zmienna do przechowywania wartosci czasu chwili poprzedniej
unsigned long previousMillis2 = 0; //zmienna do przechowywania wartosci czasu chwili poprzedniej
unsigned long currentMillis = 0;

//-----
//MAC address = "10:21:3E:47:FC:D2"; //przykładowy MAC adres, na jego podstawie wypelnic ponizej
uint8_t address[6] = {0x10, 0x21, 0x3E, 0x47, 0xFC, 0xD2};
//-----

void setup()
{
    ledcAttachPin(LED1, 1);
    ledcSetup(1, 1000, 16);
}
```

- Autoryzacja na podstawie nazwy interfejsu BT – w 99% domyślna nazwa to „V-LINK”, należy sprawdzić czy nazwa wyświetlana w liście sparowanych urządzeń z telefonem jest taka jak powyżej, jeśli jest inna należy ją zmienić w kodzie programu jak poniżej, ważna jest wielkość liter.



```
ELM_PORT.setPin("1234"); //kod pin do parowania z vlinkiem
ELM_PORT.begin("M6", true); //aktywacja BT, true=debug on

//-----
if (!ELM_PORT.connect("V-LINK"))
{
    while (count < 4) //dioda led miga 4 razy co 1 sekunde gdy nie uda sie polaczyc z interfejsem
    {
        ledcWrite(10, PWM_value_orb);
        delay(1000);
        ledcWrite(10, 0);
        delay(1000);
        count++;
    }
}
```

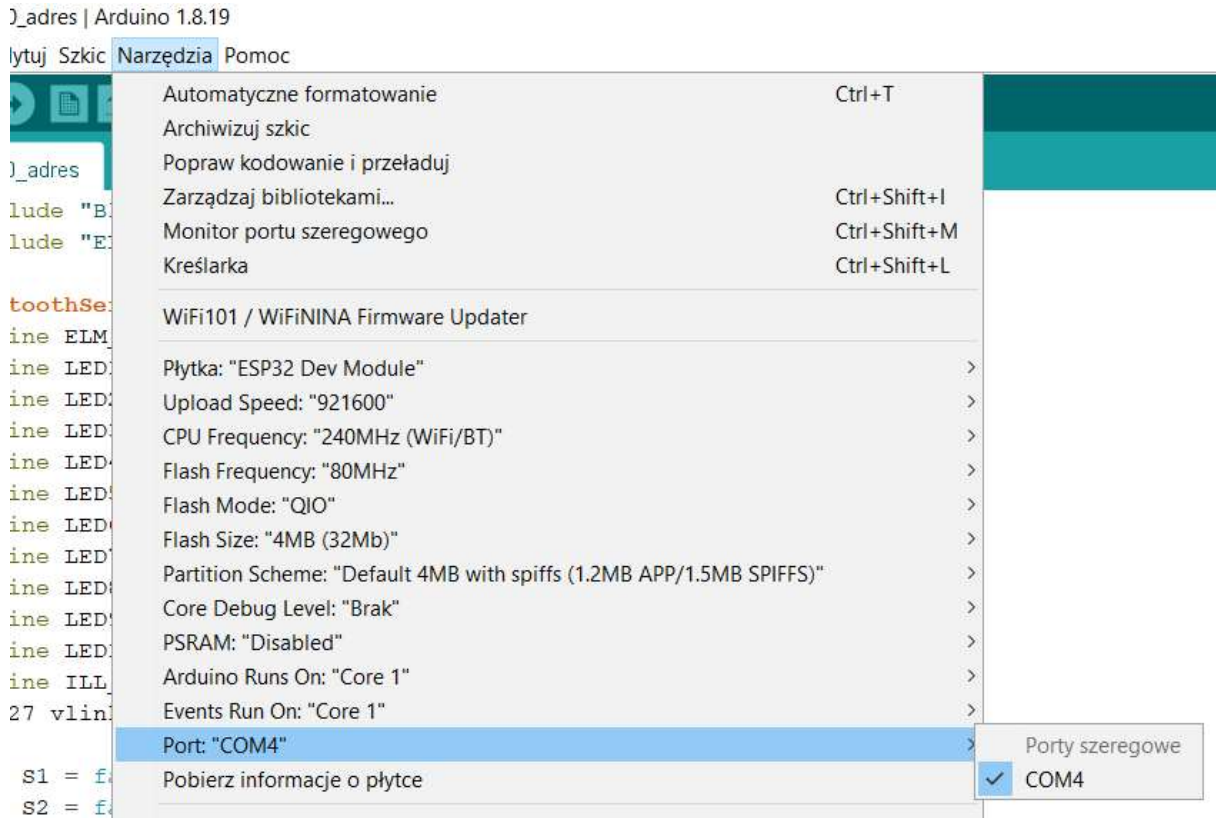


### 3. Wgrywanie kodu

Jeśli wszystko jest odpowiednio uzupełnione podłączamy nasz moduł do portu USB , czekamy aż Windows zainstaluje sterowniki, sprawdzamy czy wykonał to poprawnie, oczywiście port COM może mieć inny numer.



Ważne aby ten sam numer widniał także w opcjach Arduino IDE „Narzędzia -> Port COM”



Klikamy opcję „Wgraj”.



W pierwszym kroku środowisko kompiluje program, następnie rozpoczyna wgrywanie, widać to po przybywających kropkach w wyświetlaczu logów. Jak tylko zaczną przybywać, wciskamy fizycznie na płytce przycisk „BOOT” na 2-3 sekundy po czym puszczaemy, powinno zacząć się wgrywanie. Po zakończeniu wgrywania ostatni komunikat powinien być jak na ostatnim screenie.

```
bool S6 = false;
bool S7 = false;
bool S8 = false;
bool S9 = false;
bool S10 = false;
bool S11 = false;
bool S12 = false;
bool start_up = false;
bool ill_in_state = false;
```

<

Wgrywanie...

```
Zmienne globalne używają 33620 bajtów
esptool.py v3.3
Serial port COM4
Connecting.....
```

```
bool S6 = false;
bool S7 = false;
bool S8 = false;
bool S9 = false;
bool S10 = false;
bool S11 = false;
bool S12 = false;
bool start_up = false;
bool ill_in_state = false;
```

<

Wgrywanie...

```
Compressed 1019152 bytes to 647603...
Writing at 0x00010000... (2 %)
Writing at 0x0001a5a8... (5 %)
```

```
bool S7 = false;
bool S8 = false;
bool S9 = false;
```

<

Ładowanie zakończone.

```
Writing at 0x000fa639... (95 %)
Writing at 0x00100333... (97 %)
Writing at 0x00105bfb... (100 %)
Wrote 1019152 bytes (647603 compressed) at 0x00010000 in 8.9 seconds (effective 916.4 kbit/s)...
Hash of data verified.
```

```
Leaving...
Hard resetting via RTS pin...
```



Odłączamy USB, po 1-2 sekundach podłączamy ponownie, czekamy aż moduł zasygnalizuje niebieską diodą, że nie udało mu się nawiązać połączenia z interfejsem BT.

W kolejnym kroku należy udać się do samochodu, zainstalować interfejs BT w gnieździe, włączyć zapłon i uruchomić moduł – można poprzez kabel USB.

#### 4. Instrukcja obsługi

Po podłączeniu zasilania przez pierwszych kilka sekund układ próbuje nawiązać komunikację z modulem BT. Kiedy uda się nawiązać poprawne połączenie niebieska dioda LED (LED10) sygnalizuje to szybkim 2 krotnym szybkim miganiem diody. Jeśli za jakiegoś powodu nie uda się nawiązać połączenia sygnalizowane jest to 4 krotnym wolnym miganiem diody. Ważne jest to, że moduł próbuje nawiązać połączenie tylko w razie rozruchu, jeśli nie uda się nawiązać połączenia oznacza problem którego powód należy odnaleźć.

Wskazania oparte są o parametr PM\_GEN gdyż w normalnych warunkach przy sprawnie działającym filtrze w większości przypadków regeneracja filtra jest aktywowana wartością PM\_GEN przekraczającą 5.7-5.9. Gdyby ktoś bardzo chciał jest możliwość modyfikacji aby diody wskazywały poziom na podstawie PM\_ACC\_DSD jednak jest to bardzo niepraktyczne – wartości dość mocno skaczą w górę i w dół a finalnie i tak prawdopodobnie regenerację aktywuje wysoki poziom PM\_GEN. Cała linijka LED składa się z 10 diod LED – 1 niebieska, 4 zielone, 3 pomarańczowe, 2 czerwone. Dioda niebieska przy rozruchu służy jako informacja o stanie komunikacji BT, po uruchomieniu przejmuje rolę wskazania aktywnego wypalania. Zostaje ona uruchomiona w momencie aktywacji regeneracji filtra i pozostaje aktywna do końca cyklu wypalania, tzn. kiedy parametr dystansu od ostatniej regeneracji osiągnie

0 km. Nie uwzględnia ona przerwania regeneracji w momencie gdy pojazd porusza się w korku i zostaje przerwane wypalanie przez ECU silnika ponieważ i tak gdy tylko będzie możliwość komputer rozpocznie ponownie regenerację do momentu jej zakończenia, potwierdzonego wpisem 0 km.

Diody uruchamiane są kolejno, najpierw 4 zielone, następnie 3 pomarańczowe, w ostatniej fazie 2 czerwone. Progi dla kolejnych diod są następujące:

	< 0.30	0.30-0.92	0.93-1.55	1.56-2.18	2.19-2.81	2.82-3.44	3.45-4.07	4.08-4.70	4.71-5.33	> 5.34	REGENERACJA
LED 1		X	X	X	X	X	X	X	X	X	LED 10 + POZIOM
LED 2			X	X	X	X	X	X	X	X	LED 10 + POZIOM
LED 3				X	X	X	X	X	X	X	LED 10 + POZIOM
LED 4					X	X	X	X	X	X	LED 10 + POZIOM
LED 5						X	X	X	X	X	LED 10 + POZIOM
LED 6							X	X	X	X	LED 10 + POZIOM
LED 7								X	X	X	LED 10 + POZIOM
LED 8									X	X	LED 10 + POZIOM
LED 9										X	LED 10 + POZIOM
LED 10											LED 10 + POZIOM

Dla przykładu, jeśli wartość PM\_GEN wynosi 3.33 wtedy aktywne są diody od 1 do 5, czyli 4 zielone i jedna pomarańczowa. Podczas aktywnej regeneracji filtra świeci się dioda 10 oraz reszta w zależności od aktualnego poziomu PM\_GEN, wraz z postępem kolejne diody są wyłączane.

Układ przewiduje jedną sytuację w której ostrzega użytkownika. W momencie gdy PM\_GEN jest poniżej 4.2 przy jednoczesnym PM\_ACC\_DSD powyżej 5.7 diody od 1 do 7 świecą światłem ciągłym, natomiast diody czerwone 8 i 9 powoli migają. Taki stan jest utrzymywany do momentu aktywacji regeneracji filtra.

Można użyć wskaźnika wyłącznie jako sygnalizację wypalania, w tym celu należy podłączyć jedynie diodę LED 10, np. do jakiejś lampki/kontrolki etc.

Wskaźnik uwzględnia podłączenie do podświetlenia deski rozdzielczej tak aby wskaźnik był czytelny w słoneczny dzień oraz nie przeszkadzał podczas nocnej jazdy. Wartości natężenia można dowolnie modyfikować w programie, odpowiadają za to parametry jak poniżej edytowalne w zakresie 1-65535

```
PWM_value_g = 25;  
PWM_value_orb = 75;
```

Jest to tylko wskaźnik ułatwiający eksploatację, w przypadku problemów z filtrem należy użyć np. Forscana do weryfikacji parametrów.

Urządzenie testowane na Androidzie, współpracuje z interfejsami BT3.0, nie testowałem go z wersją 4.0 czyli BLE.